IN THE CLAIMS:

a. Please enter the following claims:

1. (Currently Amended) A method of decoding a corrupt codestream, wherein data from an underlying digital image have been partitioned into partition sets and encoded by an encoder, and wherein rules governing dependencies of said data within the partition sets are known, comprising the following steps:

observing partitions imposed by the encoder,

detecting an error in a partition set in which data that follow the error depend on said error,

analyzing the dependencies within the partition set,

determining at least one section ~~what sections~~ of encoded data that follow the error in the partition set that can be salvaged, and

decoding said sections of encoded image data in the partition set.

2. (Original) The method of claim 1, wherein said data are partitioned to allow decoding of different spatial sections of the digital image or more efficient compression for non-stationary sources.

3. (Original) The method of claim 1, wherein the dependencies are analyzed according to a state of certain switches set during encoding.

4. (Original) The method of claim 3, wherein salvageable data are determined in accordance with the state of said switches and an occurrence of the error.

5. (Original) The method of claim 4, further comprising the steps of:

determining how said error propagates through the codestream based on the state of said switches, and decoding the sections of the codestream that are not affected by the error.

6. (Original) The method of claim 5, wherein no external information on a location of the error in the codestream is available.

7. (Original) The method of claim 5, wherein external error detection provides information on a location of errors within data sections.

8. (Original) The method of claim 1, wherein the digital image is encoded in a JPEG2000 codestream.

9. (Original) The method of claim 8, wherein a sequence of digital images is encoded in a file format containing JPEG2000 codestreams.

10. (Original) The method of claim 9, wherein the file format is Motion JPEG2000

11. (Original) The method of claim 1, wherein the rules also govern dependencies between partition sets, and further comprising the step of determining what data within

dependent partition sets can be salvaged based on an occurrence of the error in the partition set.

12. (Currently Amended) A method of decoding a corrupt JPEG2000 codestream, wherein data from an underlying digital image have been partitioned into partition sets and encoded in significance propagation, magnitude refinement, and clean-up passes by a JPEG2000 encoder, and wherein rules governing dependencies within partition sets are known, comprising the following steps:

observing a partition imposed by the JPEG2000 encoder,

detecting an error in a partition set in which data that follow the error depend on said error,

analyzing the dependencies within the partition set in accordance with a state of a RESET and a BYPASS switch to determine how errors propagate through the codestream during the significance propagation, magnitude refinement, and clean-up passes, and determining at least one section ~~what sections~~ of encoded image data that follow said error in the partition set that can be salvaged, and

decoding said sections of encoded image data in the partition set.

13. (Original) The method of claim 12, wherein the partition sets are codeblocks.

14. (Original) The method of claim 13, wherein the data are further partitioned into partition sets given by packets and the rules also govern dependencies between partition sets, further comprising the step of determining what dependent partition sets can be completely or partially salvaged based on an occurrence of the error in the partition set.

15. (Original) The method of claim 12, wherein external error detection provides information on a location of said error in the codestream.

16. (Original) The method of claim 15, wherein the dependencies are analyzed and salvageable sections of encoded data determined through the steps of:

When an error occurs in the significance propagation pass of a given codeblock and RESET = False, BYPASS = False, decoding a portion of a next magnitude refinement pass within said codeblock in accordance with uncorrupt portions of κsig and σ[j];

When an error occurs in the magnitude refinement pass of a given codeblock and RESET = False, BYPASS = False, discarding all following magnitude refinement coding passes within said codeblock and decoding the two other types of coding passes;

When an error occurs in the clean-up pass of a given codeblock and RESET = False, BYPASS = False, discarding all following coding passes within said codeblock;
When an error occurs in the significance propagation pass of a given codeblock and RESET = False, BYPASS = True, decoding the next significance propagation and clean up passes partially, decoding the next magnitude refinement pass completely, decoding one more magnitude refinement pass partially, and discarding all of the following coding passes within said codeblock;

When an error occurs in the magnitude refinement pass of a given codeblock and RESET = False, BYPASS = True, continue decoding the current and all future coding passes within said codeblock;

When an error occurs in a clean-up pass of a given codeblock and RESET = False, BYPASS = True, decoding portions of the next significance propagation and magnitude

refinement passes within said codeblock in accordance with the uncorrupt portions of κsig and σ[j];

When an error occurs in the significance propagation pass of a given codeblock and RESET = True, BYPASS = False, decoding portions of all following coding passes within said codeblock in accordance with the uncorrupt portions of κsig and σ[j];

When an error occurs in the magnitude refinement pass of a given codeblock and RESET = True, BYPASS = False, continue decoding the current and all future coding passes within said codeblock;

When an error occurs in a clean-up pass of a given codeblock and RESET = True, BYPASS = False, decoding portions of all following coding passes within said codeblock in accordance with the uncorrupt portions of κsig and σ[j];

When an error occurs in the significance propagation pass of a given codeblock and RESET = True, BYPASS = True, completely decoding one more magnitude refinement pass and decoding portions of the remaining coding passes within said codeblock in accordance with the uncorrupt portions of κsig and σ[j];

When an error occurs in the magnitude refinement pass of a given codeblock and RESET = True, BYPASS = True, continue decoding the current and all future coding passes within said codeblock; and

When an error occurs in a clean-up pass of a given codeblock and RESET = True, BYPASS = True, decoding portions of all following coding passes within said codeblock in accordance with the uncorrupt portions of κsig and σ[j].

17. (Original) The method of claim 12, wherein no external information on a location of the error in the codestream is available.

18. (Original) The method of claim 17, wherein the dependencies are analyzed and salvageable sections of encoded data determined according to the steps of:

When an error occurs in the significance propagation pass of a given codeblock and RESET = False, BYPASS = False, discarding all of the following coding passes within said codeblock;

When an error occurs in the magnitude refinement of a given codeblock pass and RESET = False, BYPASS = False, discarding all following magnitude refinement coding passes and decoding the two other types of coding passes within said codeblock;

When an error occurs in a clean-up of a given codeblock pass and RESET = False, BYPASS = False, discarding all of the following coding passes within said codeblock;

When an error occurs in the significance propagation pass of a given codeblock and RESET = False, BYPASS = True, discarding all of the following significance propagation and clean-up passes and decoding the next magnitude refinement pass within said codeblock;

When an error occurs in the magnitude refinement pass of a given codeblock and RESET = False, BYPASS = True, continue decoding the current and all future coding passes within said codeblock;

When an error occurs in a clean-up pass of a given codeblock and RESET = False, BYPASS = True, discarding all of the following coding passes within said codeblock;

When an error occurs in the significance propagation pass of a given codeblock and RESET = True, BYPASS = False, discarding all of the following coding passes within said codeblock;

When an error occurs in the magnitude refinement pass of a given codeblock and RESET = True, BYPASS = False, continue decoding the current and all future coding passes within said codeblock;

When an error occurs in a clean-up pass of a given codeblock and RESET = True, BYPASS = False, discarding all of the following coding passes within said codeblock;

When an error occurs in the significance propagation pass of a given codeblock and RESET = True, BYPASS = True, discarding future significance propagation and clean-up passes and decoding one more magnitude refinement pass within said codeblock;

When an error occurs in the magnitude refinement pass of a given codeblock and RESET = True, BYPASS = True, continue decoding the current and all future coding passes within said codeblock; and

When an error occurs in a clean-up pass of a given codeblock and RESET = True, BYPASS = True, discarding all of the following coding passes within said codeblock.

19. (Original) The method of claim 12, wherein the rules further include a CAUSAL switch that govern the dependencies.

20. (Original) A method of decoding corrupt JPEG2000 codestreams where no external information on a location of errors is available, wherein data from underlying digital imagery have been partitioned into partition sets and encoded in significance propagation, magnitude refinement, and clean-up passes, and rules governing dependencies within and between said partition sets are defined by a state of RESET and BYPASS switches, comprising the steps of:

When an error occurs in the significance propagation pass of a given codeblock and RESET = False, BYPASS = False, discarding all of the following coding passes within said codeblock;

When an error occurs in the magnitude refinement of a given codeblock pass and RESET = False, BYPASS = False, discarding all following magnitude refinement coding passes and decoding the two other types of coding passes within said codeblock;

When an error occurs in a clean-up of a given codeblock pass and RESET = False, BYPASS = False, discarding all of the following coding passes within said codeblock;

When an error occurs in the significance propagation pass of a given codeblock and RESET = False, BYPASS = True, discarding all of the following significance propagation and clean-up passes and decoding the next magnitude refinement pass within said codeblock;

When an error occurs in the magnitude refinement pass of a given codeblock and RESET = False, BYPASS = True, continue decoding the current and all future coding passes within said codeblock;

When an error occurs in a clean-up pass of a given codeblock and RESET = False, BYPASS = True, discarding all of the following coding passes within said codeblock;

When an error occurs in the significance propagation pass of a given codeblock and RESET = True, BYPASS = False, discarding all of the following coding passes within said codeblock;

When an error occurs in the magnitude refinement pass of a given codeblock and RESET = True, BYPASS = False, continue decoding the current and all future coding passes within said codeblock;

When an error occurs in a clean-up pass of a given codeblock and RESET = True, BYPASS = False, discarding all of the following coding passes within said codeblock;

When an error occurs in the significance propagation pass of a given codeblock and RESET = True, BYPASS = True, discarding future significance propagation and clean-up passes and decoding one more magnitude refinement pass within said codeblock;

When an error occurs in the magnitude refinement pass of a given codeblock and RESET = True, BYPASS = True, continue decoding the current and all future coding passes within said codeblock; and

When an error occurs in a clean-up pass of a given codeblock and RESET = True, BYPASS = True, discarding all of the following coding passes within said codeblock.

21. (Original) The method of claim 20, wherein the rules further include a CAUSAL switch that governs the dependencies.

22. (Original) A method of decoding corrupt JPEG2000 codestreams where external information on a location of errors is available, wherein data from underlying digital imagery have been partitioned into partition sets and encoded in significance propagation, magnitude refinement, and clean-up passes, and rules governing dependencies within and between said partition sets are defined by a state of RESET and BYPASS switches, comprising the steps of:

When an error occurs in the significance propagation pass of a given codeblock and RESET = False, BYPASS = False, decoding a portion of the next magnitude refinement pass within said codeblock in accordance with the uncorrupt portions of κsig and $\sigma[j]$;

When an error occurs in the magnitude refinement pass of a given codeblock and RESET = False, BYPASS = False, discarding all following magnitude refinement coding passes within said codeblock and decoding the two other types of coding passes;

When an error occurs in a clean-up pass of a given codeblock and RESET = False, BYPASS = False, discarding all of the following coding passes within said codeblock;

When an error occurs in the significance propagation pass of a given codeblock and RESET = False, BYPASS = True, decoding the next significance propagation and clean up passes partially, decoding the next magnitude refinement pass completely, decoding one more magnitude refinement pass partially, and discarding all of the following coding passes within said codeblock;

When an error occurs in the magnitude refinement pass of a given codeblock and RESET = False, BYPASS = True, continue decoding the current and all future coding passes within said codeblock;

When an error occurs in a clean-up pass of a given codeblock and RESET = False, BYPASS = True, decoding portions of the next significance propagation and magnitude refinement passes within said codeblock in accordance with the uncorrupt portions of κsig and $\sigma[j]$;

When an error occurs in the significance propagation pass of a given codeblock and RESET = True, BYPASS = False, decoding portions of all following coding passes within said codeblock in accordance with the uncorrupt portions of κsig and $\sigma[j]$;

When an error occurs in the magnitude refinement pass of a given codeblock and RESET = True, BYPASS = False, continue decoding the current and all future coding passes within said codeblock;

When an error occurs in a clean-up pass of a given codeblock and RESET = True, BYPASS = False, decoding portions of all following coding passes within said codeblock in accordance with the uncorrupt portions of κsig and $\sigma[j]$;

When an error occurs in the significance propagation pass of a given codeblock and RESET = True, BYPASS = True, completely decoding one more magnitude refinement pass

and decoding portions of the remaining coding passes within said codeblock in accordance with the uncorrupt portions of κsig and σ[j];

When an error occurs in the magnitude refinement pass of a given codeblock and RESET = True, BYPASS = True, continue decoding the current and all future coding passes within said codeblock; and

When an error occurs in a clean-up pass of a given codeblock and RESET = True, BYPASS = True, decoding portions of all following coding passes within said codeblock in accordance with the uncorrupt portions of κsig and σ[j].

23. (Original) The method of claim 22, wherein the rules further include a CAUSAL switch that governs the dependencies.

24. (Currently Amended) A method of transmitting image data, comprising the steps of:

encoding the image data with a JPEG2000 compatible encoder with a RESTART switch enabled into a JPEG2000 codestream,

transmitting the JPEG2000 codestream over a channel that is susceptible to errors, and

decoding corrupted codestream with a JPEG2000 decoder that is configured to observe partitions imposed by the JPEG2000 encoder, wherein for at least one partition at least some data that follow the error depend on said error, analyze dependencies within the partition sets in accordance with a state of the RESET and BYPASS switches to determine how errors propagate through the codestream during a significance propagation, magnitude refinement, and clean-up passes, and when errors occur, determine at least one section ~~what sections~~ of encoded image data that follow the error in the partition set that can be salvaged, and decode said sections.

25. (Original) The method of claim 24, wherein the JPEG2000 codestream comprises a plurality of packets, further comprising:

storing the packets in a server,

in response to end-user requests, extracting only those additional packets of the codestream required to satisfy the end-user request,

transmitting the packets over the channel that is susceptible to errors, and decoding corrupted packets.

26. (Currently Amended) A JPEG2000 imaging system, comprising:

one or more JPEG2000 compatible image encoders used to produce content in the form of JPEG2000 codestreams,

a server that stores a JPEG2000 codestream for each image,

a distribution network that includes channels that are susceptible to errors, and

one or more display devices provided with JPEG2000 decoders that are configured to observe partitions imposed by the JPEG2000 encoder, analyze dependencies within partition sets in accordance with a state of RESET and BYPASS switches to determine how errors propagate through the codestream during a significance propagation, magnitude refinement, and clean-up passes, and when ~~errors occur~~ an error occurs in a partition in which data that follow the error depend on said error, determine at least one section ~~what sections~~ of encoded image data that follow the error in the partition set that can be salvaged and decode said sections.

27. (Original) The system of claim 26, wherein the decoder is so configured by an after-market downloadable plug-in.

28. (Currently Amended) A display device including a JPEG2000 decoder that is configured to observe the partitions imposed by a JPEG2000 encoder, analyze the dependencies within the partition sets in accordance with the state of the RESET and BYPASS switches to determine how errors propagate through the codestream during the significance propagation, magnitude refinement, and clean-up passes, and when ~~errors occur~~ an error occurs in a partition in which data that follow the error depend on said error, determine at least one section ~~what sections~~ of encoded image data that follow the error in the partition set that can be salvaged and decode those sections.

29. (Original) A display device of claim 28, wherein the decoder analyzes the dependencies between partition sets and when errors occur, determines what dependent partitions sets can be salvaged.